Driver Manual

CIF WinAC Driver

WinAC Basis and WinAC RTX

WinAC Basis 3.x/4.1, WinAC RTX 3.x/4.0/4.1/2005/2008/2009

Hilscher Gesllschaft für Systemautomation mbH

www.hilscher.com

DOC020701DRV03EN | Revision 3 | English | 2009-09 | Released | Public

Table of Contents

1 Introduction 5
1.1 List of Revisions 5
1.2 References to Additional Information 5
1.3 Legal Notes 6
1.3.1 Copyright 6
1.3.2 Important Notes 6
1.3.3 Exclusion of Liability 7
1.3.4 Export 7

2 About the Driver 8
2.1 CD Contents 8
2.2 Implementation 9
2.2.1 WinAC Basis 9
2.2.2 WinAC RTX 9
2.3 Requirements 10
2.3.1 WinAC Basis V3.x +SP1 / V4.1 10
2.3.2 WinAC RTX V3.x / V4.0 / V4.1 10
2.3.3 WinAC RTX 2005 10
2.3.4 WinAC RTX 2008 11
2.3.5 WinAC RTX 2009 11
2.4 Features 12
2.5 Limitations 12
2.6 Supported CIF Cards and Fieldbus Systems 13

3 Installation 15
3.1 WinAC Basis / Win AC RTX 15

4 STEP 7 Project Creation and Configuration 16
4.1 Install the CIF Hardware and its Device Drivers 16
4.2 Create a Fieldbus Configuration with SyCon 17
4.3 Create a new STEP7 Project 17
4.4 Configure the CIF Devices used in the STEP7 Project 19
4.5 Setup I/O Data Update Rate 21
4.6 Asynchronous Message Transfer 22
4.7 Error Checking 24

5 Fieldbus Configuration 25
5.1 Fieldbus Startup Behaviour 26
5.2 Auto Addressing 26
5.3 Handshake of the Process Data 27
5.4 User Program Monitoring 27

6 CIF-WinAC Driver 28
6.1 Driver Loading, Initialization and Removing 29
6.1.1 WinAC Basis 29
6.1.2 WinAC RTX 29
6.2 Driver Functions 30
6.3 Initialization Process 31
6.4 I/O Data Transfer between CIF Cards and Process Image 31

7 STEP 7 Program 32
7.1 Program Structure 33
7.2 Data Organization 34
7.3 Function Blocks 35
7.3.1 FB2: CIFACDriverInit 35
7.3.2 FB3: CIFACReadInput 35
7.3.3 FB4: CIFACWriteOutput 36
7.3.4 FB5: CIFACReadSlaveConfig 36
7.3.5 FB6: CIFACReadDeviceInfo 36
7.3.6 FB7: CIFACWriteMsg 37
7.3.7 FB8: CIFACReadMsg 37
7.4 Error Handling 38

© Hilscher, 2009

8 Error Numbers 39
8.1 WinAC Basis Error List 39
8.2 WinAC RTX Error List 40
8.3 CIF-WinAC Driver Error List 41
8.4 CIF Device Driver Error List 42
8.4.1 Additional Error Information 45
9 Appendix 47
9.1 List of Tables 47
9.2 List of Figures 47
9.3 Contacts 48

© Hilscher, 2009

Important Notes

Autorization:

The Hilscher CIF-WinAC driver requires a license code on the CIF hardware. An "Error –14" will be reported if there is no license code on the hardware found.

General Limitations:

- *The Hilscher CIF-WinAC driver only supports "Modular I/O Slave Devices" with not more than 60 defined modules.*

New Feature:

- The Hilscher CIF-WinAC driver now supports asynchronous fieldbus protocol services like SDOs. Please notice that this feature is restricted to the CIF-WinAC Driver for WinAC 2005, 2008 and 2009.

© Hilscher, 2009

1 Introduction

WinAC (Windows Automation Center) is a product line from the Siemens AG and designed as an integrated solution for Control, HMI, Networking and Data Processing.

WinAC is able to use a standard PC as the hardware platform to run all of it's functionality's. One of the modules is WinLC (Windows Logic Controller) which turns the PC into a SoftPLC.

Since offering an Open Development Kit (WinAC ODK, WinAC RTX ODK), it is possible to extend WinAC by custom functionality's. It is also possible to directly interact with the program scan cycle of WinLC.

These functionality's have been used to implement the Hilscher **C**ommunication **I**nter**F**aces (CIF) into WinAC Basis and WinAC RTX systems.

1.1 List of Revisions

Rev	Date	Version	Chapter	Revision
1	2002-06-24	V1.000	All	Created CIF-WinAC Driver V1.002
2	2005-11-29	V2.000	All	CIF-WinAC Basis / RTX Driver V1.200 for WinAC 3.x/4.1 CIF-WinAC RTX Driver V2.000 for WinAC RTX 2005 - Information about background I/O data processing included - Supported hardware and fieldbus table extended - Support for CIF100 cards removed
3	2009-09-02	V2.004 V3.000	2.2; 3; 4.3; 6.2; 7	CIF-WinAC RTX Driver V2.004 for WinAC RTX 2005 CIF-WinAC RTX Driver V3.000 for WinAC RTX 2008/2009 - Requirements and installation instructions for WinAC RTX 2008 and WinAC RTX 2009 included - CIFRTX driver version updated to V1.110 - Driver functions for asynchronous message transfer added

Table 1: List of Revisions

1.2 References to Additional Information

Please use the listed manuals if you are searching additional information.

Manual name	Content
CIF Device Driver	General driver description, functions and API
Protocol Manual	General protocol information and definition
Protocol Interface Manual	Protocol specific definitions and functions
CIF Toolkit Manual	General message definitions for asynchronous commands (SDOs)

Table 2: References to Additional Information

© Hilscher, 2009

1.3 Legal Notes

1.3.1 Copyright

© 2002-2009 Hilscher Gesellschaft für Systemautomation mbH

All rights reserved.

The images, photographs and texts in the accompanying material (user manual, accompanying texts, documentation, etc.) are protected by German and international copyright law as well as international trade and protection provisions. You are not authorized to duplicate these in whole or in part using technical or mechanical methods (printing, photocopying or other methods), to manipulate or transfer using electronic systems without prior written consent. You are not permitted to make changes to copyright notices, markings, trademarks or ownership declarations. The included diagrams do not take the patent situation into account. The company names and product descriptions included in this document may be trademarks or brands of the respective owners and may be trademarked or patented. Any form of further use requires the explicit consent of the respective rights owner.

1.3.2 Important Notes

The user manual, accompanying texts and the documentation were created for the use of the products by qualified experts, however, errors cannot be ruled out. For this reason, no guarantee can be made and neither juristic responsibility for erroneous information nor any liability can be assumed. Descriptions, accompanying texts and documentation included in the user manual do not present a guarantee nor any information about proper use as stipulated in the contract or a warranted feature. It cannot be ruled out that the user manual, the accompanying texts and the documentation do not correspond exactly to the described features, standards or other data of the delivered product. No warranty or guarantee regarding the correctness or accuracy of the information is assumed.

We reserve the right to change our products and their specification as well as related user manuals, accompanying texts and documentation at all times and without advance notice, without obligation to report the change. Changes will be included in future manuals and do not constitute any obligations. There is no entitlement to revisions of delivered documents. The manual delivered with the product applies.

Hilscher Gesellschaft für Systemautomation mbH is not liable under any circumstances for direct, indirect, incidental or follow-on damage or loss of earnings resulting from the use of the information contained in this publication.

1.3.3 Exclusion of Liability

The software was produced and tested with utmost care by Hilscher Gesellschaft für Systemautomation mbH and is made available as is. No warranty can be assumed for the performance and flawlessness of the software for all usage conditions and cases and for the results produced when utilized by the user. Liability for any damages that may result from the use of the hardware or software or related documents, is limited to cases of intent or grossly negligent violation of significant contractual obligations. Indemnity claims for the violation of significant contractual obligations are limited to damages that are foreseeable and typical for this type of contract.

It is strictly prohibited to use the software in the following areas:

- for military purposes or in weapon systems;
- for the design, construction, maintenance or operation of nuclear facilities;
- in air traffic control systems, air traffic or air traffic communication systems;
- in life support systems;
- in systems in which failures in the software could lead to personal injury or injuries leading to death.

We inform you that the software was not developed for use in dangerous environments requiring fail-proof control mechanisms. Use of the software in such an environment occurs at your own risk. No liability is assumed for damages or losses due to unauthorized use.

1.3.4 Export

The delivered product (including the technical data) is subject to export or import laws as well as the associated regulations of different counters, in particular those of Germany and the USA. The software may not be exported to countries where this is prohibited by the United States Export Administration Act and its additional provisions. You are obligated to comply with the regulations at your personal responsibility. We wish to inform you that you may require permission from state authorities to export, re-export or import the product.

© Hilscher, 2009

2 About the Driver

2.1 CD Contents

Folder	Content
Manuals	CIF-WinAC Driver Documentation
Sycon	Sycon Protocol Manuals
WinAC Driver	CIF-WinAC Driver Manual
Driver	CIF Driver Setup for Windows 9x/2000/XP
EDS	Electronic Datasheets for Hilscher Devices
FAQ	Frequently Asked Questions
Firmware	Firmware Files
SyCon	SyCon System Configurator
UTIL	CIF Device Utilities
COMPRO	ComPro configuration and diagnostic program
DEVVIEW	Device Viewer to indicate information about DPM
PCI	PLX Monitor to configure the PCI card
TCPIPSrv	TCP/IP Server
WinAC	CIF-WinAC Driver and Step 7 Example Projects
CIFAC Driver	CIF-WinAC Driver Setup
STEP7	Step 7 Example Projects
WinAC_3x_41	Step 7 Example Project for WinAC Base/RTX 3.x/4.1
WinAC RTX 2005	Step 7 Example Project for WinAC RTX 2005
WinAC RTX 2008	Step 7 Example Project for WinAC RTX 2008
WinAC RTX 2009	Step 7 Example Project for WinAC RTX 2009
ACROREAD\Windows	Adobe Acrobat Reader Setup (German, English, French)

Table 3: CD Contents

© Hilscher, 2009

2.2 Implementation

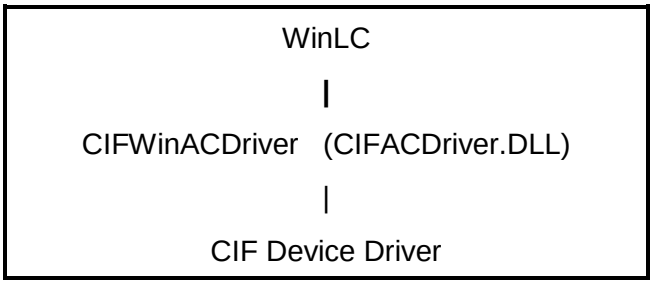
2.2.1 WinAC Basis

The implementation is based on the Siemens WinAC ODK by creating a COM object which is called from the WinLC scan cycle. The COM object is located in the file **CIFACDriver.DLL**.

Hardware access to the Hilscher CIF cards based on the “CIF Device Driver” for Windows NT 4, Windows 2000 and Windows XP.

The CIF-WinAC driver offers transparent access to the different devices. Therefore it hides the functionality’s for ISA, PCMCIA and PCI cards. The cards can be selected by the original ‘Device Type” string and a board number between 0 to 3. The board numbers can be obtained from the device driver setup program.

Overview:



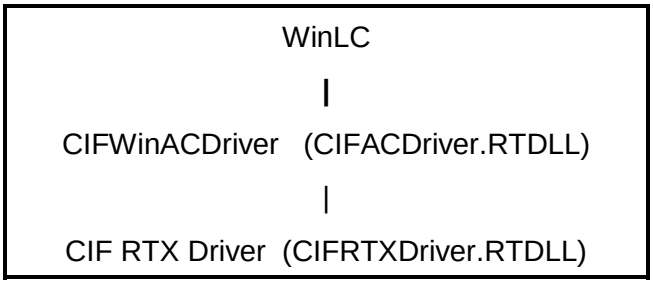
2.2.2 WinAC RTX

The implementation is based on the Siemens WinAC RTX ODK. The driver is located in the **CIFACDriver.RTDLL** which is called from the WinLC scan cycle via a calling interface.

Hardware access to the Hilscher CIF cards is based on the “CIF RTX Driver” (**CIFRTXDriver.rtdll**).

The CIF-WinAC driver offers transparent access to the different devices. Therefore it hides the functionality’s for ISA and PCI cards. The cards can be selected by the original ‘Device Type” string and a board number between 0 to 3.

Overview:



© Hilscher, 2009

2.3 Requirements

The CIF-WinAC driver is available for a wide range of SIMATIC WinAC versions. The minimum software requirements are listed below. Please notice that you need a valid Hilscher hardware license code to use a CIF Device.

2.3.1 WinAC Basis V3.x +SP1 / V4.1

Software	Version
Siemens SIMATIC WinAC Basis	V3.x SP1 / V4.1
Siemens SIMATIC Step 7	V5.0 SP3 or higher
SyCon (Fieldbus System Configurator)	V2.814 or higher
CIF-WinAC Driver for WinAC Basis V3.x SP1 / V4.1	V1.200
Hardware driver for ISA,PCMCIA,PCI cards: - CIF Device Driver V3.015 for Windows NT or - CIF Device Driver for Windows 2000/XP	 V3.015 V3.120 or higher

Table 4: Requirements - WinAC Basis V3.x +SP1 / 4.1

2.3.2 WinAC RTX V3.x / V4.0 / V4.1

Software	Version
Siemens SIMATIC WinAC RTX	V3.x / V4.0 / V4.1
Siemens SIMATIC Step 7	V5.0 SP4 or higher
SyCon (Fieldbus System Configurator)	V2.814 or higher
CIF-WinAC Driver for WinAC RTX V3.x / V4.0 / V4.1	V1.200
Hardware driver for ISA and PCI cards: - CIFRTXDriver for VentureCom RTX 4.x /5.x	 V1.106

Table 5: Requirements - WinAC RTX V3.x / 4.0 / V4.1

2.3.3 WinAC RTX 2005

Software	Version
Siemens SIMATIC WinAC RTX	2005
Siemens SIMATIC Step 7	V5.3 SP 3 or higher
SyCon (Fieldbus System Configurator)	V2.814 or higher
CIF-WinAC Driver for WinAC RTX 2005	V2.004
Hardware driver for ISA,PCMCIA,PCI cards: - CIFRTXDriver for Ardence RTX 6.5	 V1.110

Table 6: Requirements - WinAC RTX 2005

© Hilscher, 2009

2.3.4 WinAC RTX 2008

Software	Version
Siemens SIMATIC WinAC RTX	2008
Siemens SIMATIC Step 7	V5.4 SP 3 or higher
Hilscher SyCon (Fieldbus System Configurator)	V2.814 or higher
CIF-WinAC Driver for WinAC RTX 2008/2009	V3.000
Hardware driver for ISA,PCMCIA,PCI cards: - CIFRTXDriver for IntervalZero RTX 8.1	 V1.110

Table 7: Requirements - WinAC RTX 2008

2.3.5 WinAC RTX 2009

Software	Version
Siemens SIMATIC WinAC RTX	2009
Siemens SIMATIC Step 7	V5.4 SP 4 or higher
STEP 7 Hardware Update (HSP 192)	V1.0
Hilscher SyCon (Fieldbus System Configurator)	V2.814 or higher
CIF-WinAC Driver for WinAC RTX 2008/2009	V3.000
Hardware driver for ISA,PCMCIA,PCI cards: - CIFRTXDriver for IntervalZero RTX 8.1	 V1.110

Table 8: Requirements - WinAC RTX 2009

© Hilscher, 2009

2.4 Features

WinAC Basis V3.x / 4.0 / 4.1:

- Hardware detection and setup for ISA, PCMCIA, PCI by standard driver functions

WinAC RTX (all versions):

- Automatic hardware detection for ISA and PCI hardware by the CIF RTX driver
- Asynchronous fieldbus protocol services (CIF-WinAC Driver V2.004 and higher)

General:

- Device selection by name and board number
- All supported devices can be used simultaneously
- Up to four devices possible
- Automatic firmware comparison and download during startup possible
- Automatic configuration comparison and download during startup possible
- Unique device selection of the hardware by serial and device number possible including automatic card replacement detection
- Fieldbus configuration by SyCon, no fieldbus configuration in STEP7 needed
- Slave I/O data are mapped 1:1 into STEP7 data blocks
- I/O data update with hardware processed in back round thread

2.5 Limitations

WinAC Basis V3.x / 4.0 /4.1:

- Only master cards are supported yet
- Only one PCMCIA card at a time supported under Windows NT 4

WinAC RTX (all versions):

- Only master cards are supported yet
- PCMCIA cards are not supported

© Hilscher, 2009

2.6 Supported CIF Cards and Fieldbus Systems

The following table shows the currently supported CIF cards and fieldbus systems.

Fieldbus System	Device Type	Firmware Name	Firmware File Name	Device Driver
CANopen	COM-COM	CANopen COM-COM	COM.H5E	CIF Device Driver
	COM-CA-COM	CANopen COMCCOM	COMCOMC.E02	
	COMCCOM	CANopen COMCCOM	COMCOMC.E02	
	CIF30CAN	CANopen CIF30CAN	COM.H36	
	C104-CAN	CANOpen C104-CAN	COM.H7N	
	C104PCOM	ANOpen C104PCOM	COM104P.E02	
	CIF50CAN	CANopen CIF50CAN	COM.H66	
	CIF60CAN	CANopen CIF60CAN	COM.H65	
	CIF80COM	CANopen CIF80COM	COM.E02	
	PMCCOM	CANopen PMCCOM	COMPMC.E02	
	CIF60CAN	CANopen CIF60CAN	COM.H65	
DeviceNet	COM-DNM	DNM COM-DNM	DNM.H5F	CIF Device Driver
	COM-CA-DNM	DNM COMCDNM	DNMCOMC.E03	
	COMCDNM	DNM COMCDNM	DNMCOMC.E03	
	CIF30DNM	DNM CIF30DNM	DNM.H38	
	C104-DNM	DNM C104-DNM	DNM.H7L	
	C104PDNM	DNM C104PDNM	DNM104P.E03	
	CIF50DNM	DNM CIF50DNM	DNM.H68	
	CIF50DNM	DNM CIF50DNM	DNMC50.E2A	
	CIF60DNM	DNM CIF60DNM	DNM.H65	
	CIF60DNM	DNM CIF60DNM	DNMC60.E00	
	CIF80DNM	DNM CIF80DNM	DNMC80.E03	
	PMC	DNM PMC	DNMPMC.E03	
	PMCDNM	DNM PMCDNM	DNMPMC.E03	
InterBus	COM-IBM	IBM COM-IBM	IBM.H53	CIF Device Driver
	CIF30IBM	IBM CIF30IBM	IBM.H35	
	C104IBM	IBM C104IBM	IBM.H78	
	C50IBM	IBM C50IBM	IBM.H69	
	C60IBM	IBM C60IBM	IBM.H6B	

Table 9: Supported CIF Cards and Fieldbus Systems (Part 1)

Continued on next page.

© Hilscher, 2009

Fieldbus System	Device Type	Firmware Name	Firmware File Name	Device Driver
PROFIBUS	CPS1-DPM	DPM CPS1-DPM	DPM.X52	CIF Device Driver
	COM-DPM	DPM COM-DPM	DPM.H58	
	COM-PB	PB-COMBICOM-PB	DPM.H58	
	COM-CA-DPM	DPM COMCADPM	DPMCOMCA.E01	
	CIF30PB	PB-COMBICOM-PB	PB.H5C	
	CIF30PB	PB-COMBICIF30PB	PB.H32	
	CIF30DPM	DPM CIF30DPM	DPM.H33	
	CIF104DP	DPM CIF104DP	DPM.H73	
	C104DPMR	DPM C104DPMR	DPM.H7B	
	C104-PB	PB-COMBIC104-PB	PB.H7C	
	C104PBR	PB-COMBIC104PBR	PB.H7B"	
	C104PBE	PB-COMBIC104PBE	PB.H7U	
	C104P-PB	PB-COMBIC104P-PB	PBC104P.E01	
	CIF50-PB	PB-COMBICIF50-PB	PB.H62	
	CIF50-PB	PB-COMBICIF50-PB	PBC50.E2B	
	CIF60PB	PB-COMBICIF60PB	PB.H61	
	CIF60PB	PB-COMBICIF60-PB	PBC60.E01	
	CIF80-PB	PB-COMBICIF80-PB	PBC80.E01	
	PMC-PB	PB-COMBIPMC-PB	PBPMC.E01	

Table 10: Supported CIF Cards and Fieldbus Systems (Part 2)

© Hilscher, 2009

3 Installation

All Hilscher software components including their documentation are provided on a single product CD. Following components are located on this CD:

- **SyCon** (System Configurator)

- CIF-WinAC Driver V1.200 for WinAC Basis/WinAC RTX 3.x / 4.0 / 4.1
- CIF Device Driver and its components
- CIF RTX Driver V1.106 for VenturCom RTX V4.3.2.1 / 5.x

- CIF-WinAC Driver V2.004 for WinAC RTX 2005
- CIF RTX Driver V1.110 for Ardence RTX 6.5

- CIF-WinAC Driver V3.000 for WinAC RTX 2008/2009
- CIF RTX Driver V1.110 for IntervalZero RTX 8.1

NOTE: The Hilscher product CD does not include any Siemens components. Please contact Siemens how to obtain their products and how to install them. We suggest to first install the Siemens software.

The Hilscher installation procedure at first runs the SyCon installation. Afterwards it starts the installation of the WinAC driver and the CIF device drivers. Each installation provides product specific options like a standard destination directory which can be changed. All components will be copied to the selected destination directories and their necessary registration procedures are processed.

3.1 WinAC Basis / Win AC RTX

Step1: Install STEP7 like described by Siemens

Step2: Install WinAC Basis or RTX like described by Siemens

Step3: Start the setup program from the Hilscher product CD

© Hilscher, 2009

4 STEP 7 Project Creation and Configuration

This chapter describes how to create a new STEP 7 project including Hilscher fieldbus support.

Step1: Install and configure the CIF hardware and the necessary device drivers

Step2: Create a fieldbus configuration with SyCon

Step3: Create a new STEP7 project and use the FBs and DBs from the example program. Or change the existing example program CIFACBasisProg, CIFACRtxProg, CIFACRtx2005Prog, CIFACRtx2008Prog, CIFACRtx2009Prog depending to the runtime environment

Step4: Configure the CIF devices used by the STEP7 project

4.1 Install the CIF Hardware and its Device Drivers

Please consult the Hilscher product CD inlet and the device driver manual (DRVDEV.PDF) how to correctly install and configure the CIF hardware and its device drivers.

WinAC Basis:

Install the CIF hardware into the PC and the Windows NT / Windows 2000 or Windows XP device driver which are necessary for the WinAC Basis system. Both SyCon and the CIF-WinAC Driver requires the hardware drivers to access the CIF hardware.

WinAC RTX:

In a WinAC RTX system, the Windows device drivers are only used by SyCon to directly download a configuration to the CIF hardware and to run SyCon debug functionality's. The CIF-WinAC Driver uses only the CIFRTXDriver to access the hardware.

ATTENTION: The Windows NT / Windows 2000 / Windows XP device driver must be deactivated as soon as the WinAC runtime system is activated.
Otherwise it is possible that Win32 programs are also able to access the CIF hardware which results in unexpected system and fieldbus behaviour.
- On Windows NT based systems check the Windows NT ***'Control Panel-Devices'*** entry **CIFDEV** and deactivate the driver.
- On Windows 2000 / Windows XP based systems you have to use a RTX Tool to assign the hardware to RTX. Please consult the WinAC RTX documentation how this is done.

© Hilscher, 2009

4.2 Create a Fieldbus Configuration with SyCon

Contact the SyCon manual how to create a valid fieldbus configuration.

NOTE: SyCon allows to export the configuration into a DBM file which can be later used in a STEP7 program to be checked against an already on the hardware existing configuration and to download it if they are not equal.

4.3 Create a new STEP7 Project

Please consult the STEP7 documentation how to use STEP7.

Step1: Create a new STEP7 project

Step 2: Include a **"SIMATIC PC-Station"**

Step 3: Change to the hardware configuration open the hardware catalog, select ***"SIMATIC PC Station"*** and insert, from the entry ***"Controller"*** the ***"WinLC"*** or ***"WinLC RTX"*** entry into the configuration.
The driver does not need any information from the STEP7 hardware configuration, therefore answer all following questions with ***"NO"***.

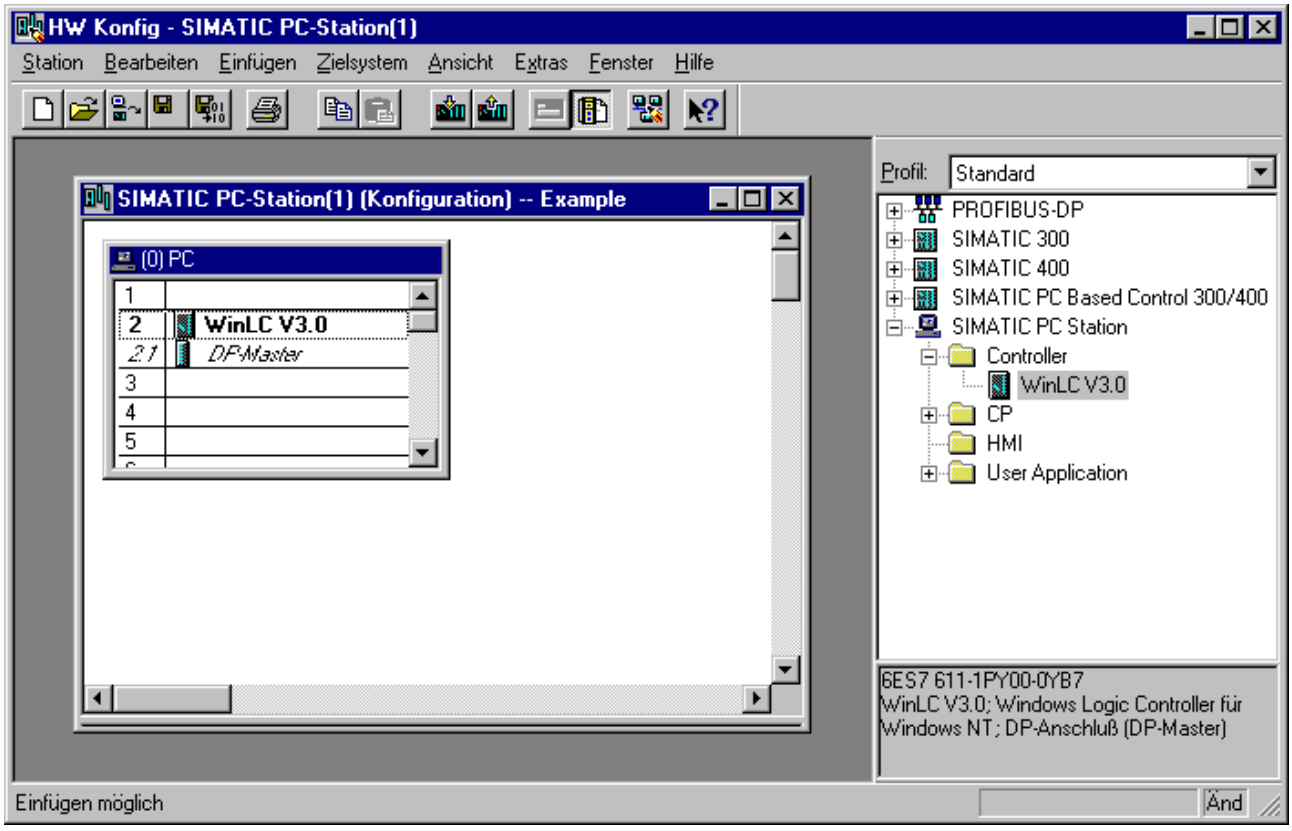


Figure 1: STEP7 Hardware Configuration

NOTE: You will see WinLC RTX when working in a WinAC environment.

© Hilscher, 2009

Step4: Change to the program folder in the STEP7 project and insert OB1, OB100, SFB65001 and SFB65002 and all FBs and DBs from the example project. Insert also the symbols from the example project

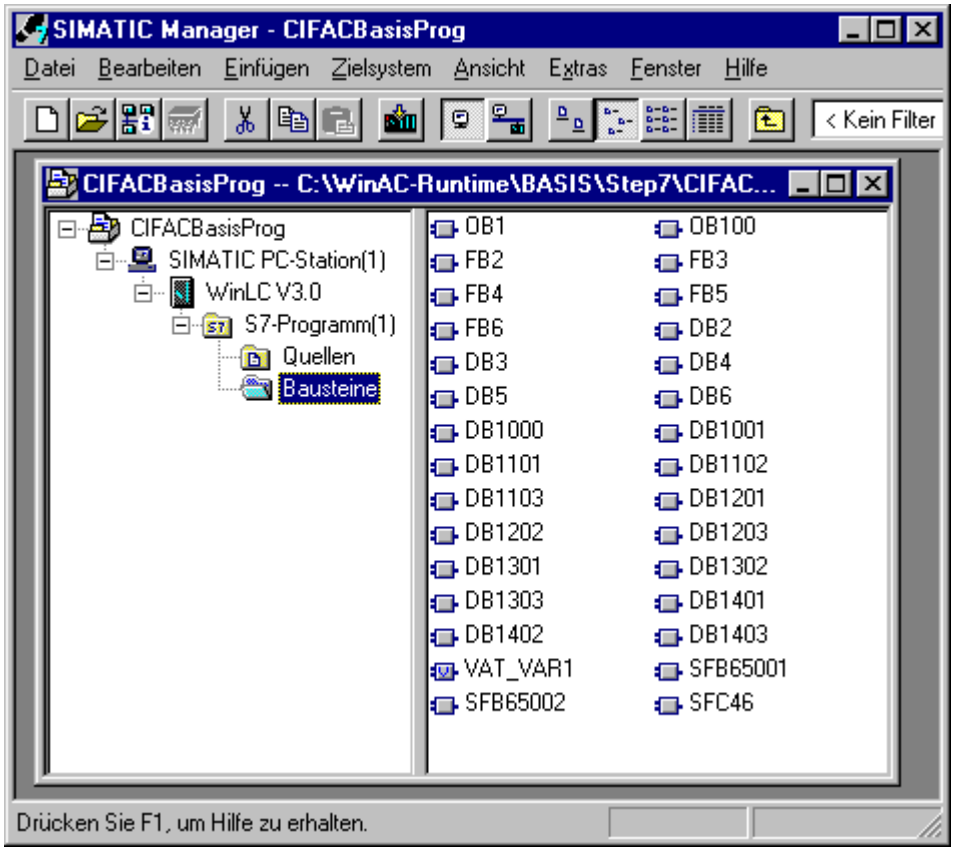


Figure 2: STEP7 Project View

NOTE: Make sure to use the example program which corresponds to the WinAC system (WinAC Basis –> CIFACBasisProg, WinAC RTX -> CIFACRtxProg, WinAC RTX 2005 -> CIFACRtx2005Prog, WinAC RTX 2008 -> CIFACRtx2008Prog, WinAC RTX 2009 -> CIFACRtx2009Prog) otherwise SFB65001 and SFB65002 are not working correctly.

After creating a STEP7 project the CIF-WinAC Driver must be configured. The configuration takes place in the DB_CONFIG (DB2) and has to match the installed CIF hardware.

© Hilscher, 2009

4.4 Configure the CIF Devices used in the STEP7 Project

The configuration of the CIF-WinAC Driver and the used CIF hardware takes place in DB_CONFIG. All other FBs are based on this data block. DB_CONFIG is the instance data block of FB2.

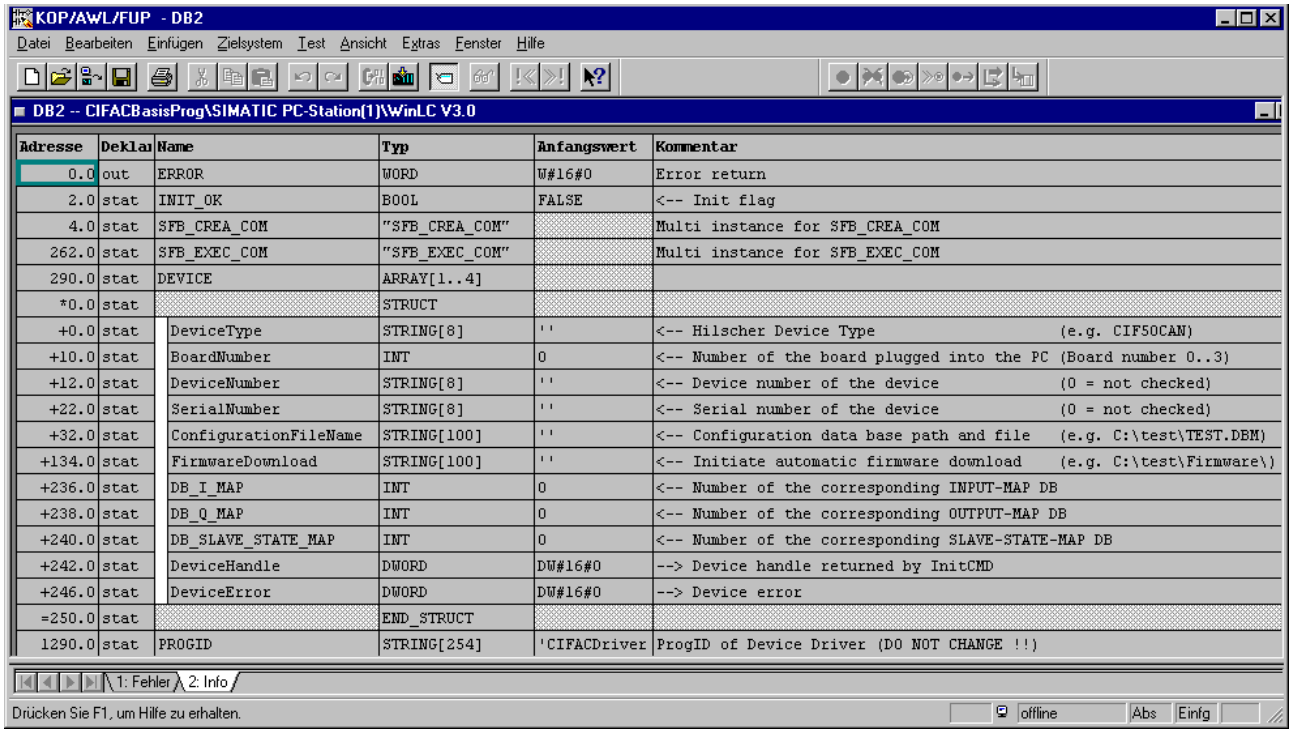

KOP/AWL/FUP - DB2
Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe
DB2 -- CIFACBasisProg\SIMATIC PC-Station(1)\WinLC V3.0
Adresse Deklar Name Typ Anfangswert Kommentar
0.0 out ERROR WORD W#16#0 Error return
2.0 stat INIT_OK BOOL FALSE <-- Init flag
4.0 stat SFB_CREA_COM "SFB_CREA_COM" Multi instance for SFB_CREA_COM
262.0 stat SFB_EXEC_COM "SFB_EXEC_COM" Multi instance for SFB_EXEC_COM
290.0 stat DEVICE ARRAY[1..4]
*0.0 stat STRUCT
+0.0 stat DeviceType STRING[8] '' <-- Hilscher Device Type (e.g. CIF50CAN)
+10.0 stat BoardNumber INT 0 <-- Number of the board plugged into the PC (Board number 0..3)
+12.0 stat DeviceNumber STRING[8] '' <-- Device number of the device (0 = not checked)
+22.0 stat SerialNumber STRING[8] '' <-- Serial number of the device (0 = not checked)
+32.0 stat ConfigurationFileName STRING[100] '' <-- Configuration data base path and file (e.g. C:\test\TEST.DBM)
+134.0 stat FirmwareDownload STRING[100] '' <-- Initiate automatic firmware download (e.g. C:\test\Firmware\)
+236.0 stat DB_I_MAP INT 0 <-- Number of the corresponding INPUT-MAP DB
+238.0 stat DB_Q_MAP INT 0 <-- Number of the corresponding OUTPUT-MAP DB
+240.0 stat DB_SLAVE_STATE_MAP INT 0 <-- Number of the corresponding SLAVE-STATE-MAP DB
+242.0 stat DeviceHandle DWORD DW#16#0 --> Device handle returned by InitCMD
+246.0 stat DeviceError DWORD DW#16#0 --> Device error
=250.0 stat END_STRUCT
1290.0 stat PROGID STRING[254] 'CIFACDriver ProgID of Device Driver (DO NOT CHANGE !!)
1: Fehler 2: Info
Drücken Sie F1, um Hilfe zu erhalten.
offline Abs Einfg

Figure 3: STEP7 Declaration View of DB_CONFIG

Data description:

Error:

Error return value for FB2

INIT_OK:

Local initialization state (must be cleared on beginning of OB100)

SFB_CREA_COM/SFB_EXEC_COM:
WinAC interface to access third party software. Accessed via multi instance.

DEVICE-ARRAY[1..4]:
Device structure, allows up to four CIF devices at the same time.

DeviceType:

Name of the CIF device like defined in the *'Supported Devices Table'* (e.g. CIF50CAN).

BoardNumber:

Can be obtained from the CIF device driver if WinAC Basis is used. On WinAC RTX, the CIF-WinAC Driver automatically enumerates the devices. In this case, board numbers are counted independent from 0 to 3 for ISA and PCI cards.

© Hilscher, 2009

DeviceNumber/SerialNumber:
Are used to unambiguous identify a specific CIF card. This could be necessary if more than one CIF card of the same fieldbus type is used. Both values must be defined. The values can be found on the CIF card label.

ConfigurationFileName:
If not and empty string, it defines the SyCon configuration file (DBM file) which should be used on the device. If configured, the CIF-WinAC Driver will check the configuration against an existing configuration on the hardware and download it if they are not equal. The give name must include the file path, file name and file extension (e.g. **"C:\TestFiles\Test.dbm"**).

FirmwareDownload:
If not an empty string, it defines the file path where the driver will find the firmware file for the given CIF card. The CIF-WinAC driver already knows the file name, which is suitable for the hardware. Only the path to the file must be defined (e.g. **"C:\TestFiles\Firmware\"**). The file path must include the trailing **"\"** character.

DB_I_MAP:
Defines the data block number to store the INPUT process data image of this device. The example program defines a size of 3584 byte. This can be reduced to the real used size. The CIF-WinAC driver will checked the defined size an returns with an error if it is not sufficient.

DB_Q_MAP:
Defines the data block number which holds OUTPUT process data image of this device. The example program defines a size of 3584 byte. This can be reduced to the real used size. The CIF-WinAC driver will checked the defined size an returns with an error if it is not sufficient.

DB_SLAVE_STATE_MAP:
Defines the data block number to store the slave state information of this device.

DeviceHandle:
Generated by the CIF-WinAC Driver during device initialization. Do not change this value.

DeviceError:
Used to store device errors and to setup.
It is also used during startup to change the cycle time of the backround I/O data thread. If this value is not 0 during startup, the Value is used as the I/O data update rate in milliseconds.

PROGID/INSTANCEID:
This parameter is necessary for WinAC and tells the system which driver to start for SFB65001 (SFB_CREA_COM). The value is different for WinAC Basis ('**CIFACDriver.CIFACDriver**') and WinAC RTX ('***RTSS:CIFRTXDriver.rtdll**').

© Hilscher, 2009

4.5 Setup I/O Data Update Rate

The CIF-WinAC driver does not directly access the I/O image from the hardware. The I/O data are buffered in an internal I/O buffer area.

For the I/O data update with the hardware, the driver generates a so called background thread for each device. The standard update rate is 2 milliseconds.

Default I/O Data Update Rate is 2 milliseconds

The update rate for each device can be changed via the **DeviceError** value in the global configuration block **DB_CONFIG**.

If **DeviceError** is 0 during the startup of the Step 7 program, the default update rate is is used. A value unequal to 0 is used as a new update rate given in milliseconds.

I/O Update Rate Settings	
DeviceError = 0	Default update rate = 2 ms
DeviceError > 0	New update rate in milliseconds (e.g. DeviceError = 10 => Update Rate = 10 ms)

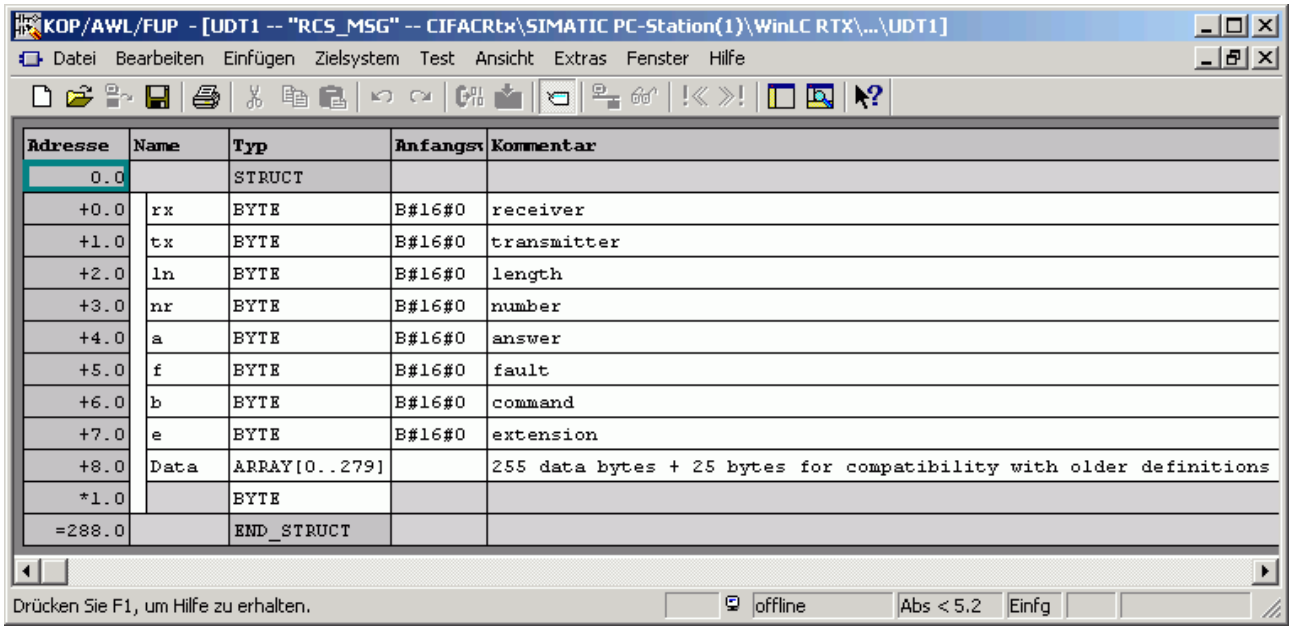
ATTENTION: Depending of the fieldbus system, the configuration and the number of connected slave devices, this update rate can be too fast. In such a case the Read/Write function of the CIF-WinAC driver returns an error **-22** (**DEVICE IO data exchange failed**).

© Hilscher, 2009

4.6 Asynchronous Message Transfer

With release of the CIF-WinAC driver version V2.004 the Hilscher CIF-WinAC driver supports asynchronous fieldbus protocol services like SDOs. Writing a asynchronous message to the CIF hardware is done via the data block DB_MSG_WRITE. Incoming messages are stored in the data block DB_MSG_READ. The declaration of both data blocks correspond to the structure of the RCS message format. A description of the message data block is given below.

NOTE: CIF cards are working with "Little Endian" data representation while STEP7 uses a "Big Endian" data representation.
I/O data are converted by the WinAC driver to the STEP 7 data representation.
Message data are not converted, because the data content depends on the message command. Converting the message data into the correct data representation is part of the user application.


KOP/AWL/FUP - [UDT1 -- "RCS_MSG" -- CIFACRtx\SIMATIC PC-Station(1)\WinLC RTX\...\UDT1]
Datei Bearbeiten Einfügen Zielsystem Test Ansicht Extras Fenster Hilfe
Drücken Sie F1, um Hilfe zu erhalten.
offline
Abs < 5.2
Einfg

Adresse	Name	Typ	Anfangsw	Kommentar
0.0		STRUCT		
+0.0	rx	BYTE	B#16#0	receiver
+1.0	tx	BYTE	B#16#0	transmitter
+2.0	ln	BYTE	B#16#0	length
+3.0	nr	BYTE	B#16#0	number
+4.0	a	BYTE	B#16#0	answer
+5.0	f	BYTE	B#16#0	fault
+6.0	b	BYTE	B#16#0	command
+7.0	e	BYTE	B#16#0	extension
+8.0	Data	ARRAY[0..279]		255 data bytes + 25 bytes for compatibility with older definitions
*1.0		BYTE		
=288.0		END_STRUCT		

Figure 4: STEP7 Structure of a RCS Message

© Hilscher, 2009

Data description:

rx: Token of the receiver task

tx: Token of the transmitter task

ln: Length of the user data.

nr: Sequential number of the message.

a: Answer message code.

f: Error code.

b: Command message code.

e: Message extension.

Data[]: User data.

For detailed information about RCS messages please consult the manual 'CIF Toolkit (tke.pdf)' and for the protocol specific message commands the corresponding protocol interface manual (e.g. DPM_pie.pdf =>PROFIBUS-DP Master protocol interface manual).

© Hilscher, 2009

4.7 Error Checking

CIFAC_Status_Word (MW 60) is a global word which will receive error values. If the value is 0 after returning from a FB call, no error occurred. Value = 1 indicates device specific errors. These errors are inserted into the value *'DeviceError'* located in the device array in DB_CONFIG. Error numbers > 1 are defined as global errors.

NOTE: OB100 always stops on all errors from FB2. Also if two fieldbus systems are installed and one of them can't be started. The user is responsible to change the error handling in OB100 to allow a system to start if one of the defined master devices or fieldbus systems does not work.

© Hilscher, 2009

5 Fieldbus Configuration

This section describes how to configure a CIF master device

NOTE: Please consult the SyCon user manual for a detailed description how to create a valid fieldbus configuration and how to change the fieldbus settings.

A fieldbus configuration must be created by SyCon (**Sy**stem **Con**figurator). After creating a valid configuration SyCon is able to either download the configuration directly to the CIF hardware or export the configuration into a database file (.DBM). This exported file can be later used to run the automatic database checking and download.

SyCon also provides a wide range of options and fieldbus specific settings. But not all of the settings are useful in a PLC environment. Because of this, the SyCon installation provides some preset values which are important.

Preset values:

- Addressing mode: **Auto addressing enabled**
- Startup behaviour: Controlled Release of the Communication
- Handshake of the Process Data: Buffered host controlled
- User program monitoring: 1000 milliseconds

© Hilscher, 2009

5.1 Fieldbus Startup Behaviour

SyCon allows to configure the start up behaviour of a fieldbus system.

It can be '*Automatic release of the communication by the device*', which means the master starts the fieldbus system as soon as the card has finished it's power on sequence. The fieldbus becomes active and slaves are able to driver there outputs.

The option '*Controlled release of the communication by the application program'* can be used to prevent an automatic startup of the field bus system. This enables the application to be fully initialized before starting the fieldbus communication.

Fieldbus Startup Behaviour	
'Automatic release of the communication by the device'	The fieldbus system will start up as soon as the card has finished it's power on sequence
'*Controlled release of the communication by the application program*' (**recommended, default**)	The start of the fieldbus system can be controlled by an application

NOTE: The WinAC driver always starts without starting the fieldbus system. Therefore an explicit start function must be executed within the application.

5.2 Auto Addressing

SyCon allows to setup the way how the slave process data are sorted into the process data image of the master device.

'Auto addressing enabled' forces SyCon to place the slave process data, in an increasing order , starting at offset 0, into the process data image of the master device. The slaves are processed in the order like they are inserted into the fieldbus configuration.

'Auto addressing disabled' allows the user to manually assign the process data position for each slave in the process data image of the master device.

SyCon provides the possibility to review the created address table.

Auto Addressing	
*'Auto addressing ENABLED' (**default**)*	SyCon automatically places the slave process data into the master process data image, starting at offset 0
'Auto addressing DISABLED'	User manually place the slave process data into the masters process data image

NOTE: Consult the SyCon manual about a closer description

© Hilscher, 2009

5.3 Handshake of the Process Data

Handshake modes are used to control the access to the I/O process data image between the PC (Host) and the CIF card (Device). The setting of the transfer mode is very important because it will influence the consistency of the process data and the fieldbus behaviour. Transfer modes are only available on master cards.

The modes are configured via SyCon which offers up to six modes, depending on the used CIF hardware and fieldbus system.

Handshake Mode	Supported
'Bus synchronous, device controlled'	**NO** (Only possible on dedicated systems, because the system must respond during a bus data cycle which can be less than 350 micro seconds)
'Buffered device controlled'	**YES**
'No consistence, uncontrolled'	**YES** (Not recommended, process data which are not of the type byte can be transferred inconsistent)
'Buffered host controlled' (**recommended, default**)	**YES**
'Bus synchronous, host controlled'	**YES** (Not on all fieldbus systems and CIF cards available. System is responsible to drive the bus)
'Buffered, extended host controlled'	**NO**

5.4 User Program Monitoring

Under '*User Program Monitoring*', SyCon allows to setup the device watchdog time. This time is used to shout down the fieldbus system in case of an abnormal program behaviour.

User Program Monitoring	
'Watchdog time'	Timeout until the fieldbus will be stopped in case of an abnormal program behaviour. 1000 ms = Default 0 ms = Function disabled

© Hilscher, 2009

6 CIF-WinAC Driver

The following section describes the internal driver functions.

The WinAC Basis driver is based on the **CIF Device Driver** interface (Cif32DLL.DLL). The WinAC RTX driver is based on the **CIF RTX Driver** (CIFRTXDriver.rtdll).

WinAC Basis and WinAC RTX are using a different way to call the driver. The basis version uses a COM interface and the RTX version uses a calling interface. The connection between the STEP7 part of the program and the CIF-WinAC Driver is done by two SFBs.

SFB65001 (CREA_COM) creates/loads the CIF-WinAC Driver. CREA_COM uses the parameter PROGID/INSTANCEID to define the used program. On WinAC Basis the parameter must be defined as *'CIFACDriver.CIFACDriver'* and WinAC RTX needs the entry *'*RTSS:CifACRTXDriver.rtdll'*.

SFB65002 (EXEC_COM) is used to access the driver functions. It uses a handle to the driver created by CREA_COM.

© Hilscher, 2009

6.1 Driver Loading, Initialization and Removing

The CIF-WinAC Driver will be called during different WinAC states to carry out its functions.

6.1.1 WinAC Basis

WinAC State	Description
Loading of WinLC	During the start of WinLC (Logic Controller) the COM interface to the driver will be instanciated. At this time the driver object is generated and the APIs of the underlying hardware drivers are loaded by LoadLibrary().
Unloading of WinLC	Unloading of the Logic Controller will remove the COM interface which also closes the hardware drivers, frees API DLLs and destroys the driver object.
Activate	DevOpenDriver() of the loaded hardware drivers will be called
Deactivate	DevCloseDriver() of the loaded hardware drivers will be called
Execute	Calls the driver functions, described in the section *'Driver Functions'*

Table 11: Driver Loading, Initialization and Removing – WinAC Basis

6.1.2 WinAC RTX

WinAC State	Description
ODKCreate	During the start of WinLC (Logic Controller) the COM interface to the driver will be instanciated. At this time the driver object is generated and the APIs of the underlying hardware drivers are loaded by LoadLibrary().
ODKRelease	Unloading of the Logic Controller will remove the COM interface which also closes the hardware drivers, frees API DLLs and destroys the driver object.
Activate	DevOpenDriver() of the loaded hardware drivers will be called
Deactivate	DevCloseDriver() of the loaded hardware drivers will be called
Execute	Calls the driver functions, described in the section *'Driver Functions'*

Table 12: Driver Loading, Initialization and Removing – WinAC RTX

© Hilscher, 2009

6.2 Driver Functions

Currently implemented driver functions:

Command	Code	Description	Underlying Device Driver Functions
CIF_CMD_INIT	1	Initialization of the specified CIF devices. Creation of the I/O data exchange background process.	- DevInitBoard() - DevGetBoardInfoEx() - DevExtendedData() - DevDownload() - DevReset() - DevSetHostState()
CIF_CMD_CONFIG	2	Read the configuration of one slave device	Reads the station description of a specific slave device
CIF_CMD_WRITE_IO	3	Write OUTPUT data to a CIF device	- DevTriggerWatchDog() - DevExchangeIOErr()
CIF_CMD_READ_IO	4	Read INPUT data from a CIF device	- DevTriggerWatchDog() - DevExchangeIOErr()
CIF_CMD_READ_STATE	6	Read task state information	Returns the information from COMSTATE structure read by - DevExchangeIOErr()
CIF_CMD_WRITE_MSG	7	Write a message to a CIF device	- DevPutMessage()
CIF_CMD_READ_MSG	8	Read a message from a CIF device	- DevGetMessage()
CIF_CMD_START_COM	9	Start fieldbus communication	- DevSetHostState()
CIF_CMD_STOP_COM	10	Stop fieldbus communication	- DevSetHostState()
CIF_CMD_DEVICE_INFO	12	Read global information about a CIF device	- DevGetInfo()

Table 13: Driver Functions

© Hilscher, 2009

6.3 Initialization Process

The following work will be carried out by the driver during initialization:

Hardware searching and setup	Done by the **CIF Device Driver** and its tools for the basis version. Automatically by the **CIF RTX Driver** for the RTX version.
Hardware reset	Each CIF card will be restarted by processing a cold reboot. This will be done to get boards working which have been previously stopped by a watchdog hit.
Firmware download	If the application provides a firmware file directory, the driver will check the firmware file version against the version currently running on the CIF card. If the they are different, the driver carries out an automatic firmware download.
Configuration download	If the application provides a configuration database file directory and name, the driver will check the configuration database against the database currently located on the CIF card. If the databases are different, the driver carries out an automatic database download.
Serial/Device number checking	If a device and serial number is provided by application, the driver will check these against the information on the CIF card.
License code checking	The driver will return error –14 if a CIF card does not have a license code.
Fieldbus Communication	The fieldbus communication will be always stopped before leaving the driver.

After the initialization procedure the CIF card is ready to start..

6.4 I/O Data Transfer between CIF Cards and Process Image

During the initialization process, the driver automatically reads the slave configuration from the master device. It uses the slave information to calculate the maximum amount of input and output data which must be transferred between the hardware and the program buffers.

The amount of data are always calculated starting at input and output offset 0.

© Hilscher, 2009

7 STEP 7 Program

This section describes the STEP 7 program, which is part of the WinAC implementation.

Two versions of the program are existing. One for the **WinAC Basis** and one for **WinAC RTX** system. The only differences between the two programs are SFB65001 and SFB 65002, and their initialization, which depends on the target system.

Program Name:	WinAC Basis	CIFACBasisProg	(CIFACBas)
	WinAC RTX	CIFACRTXProg	(CIFACRtx)
	WinAC RTX 2005	CIFACRTX2005Prog	(CIFACRtx)
	WinAC RTX 2008	CIFACRTX2008Prog	(CIFACRtx)
	WinAC RTX 2009	CIFACRTX2009Prog	(CIFACRtx)

The program allows to run up to four master devices with the same or different fieldbus systems simultaneously.

All necessary configuration data and runtime states are placed into one single data block named DB_CONFIG (DB2). This data block contains the instances for SFB65001 and SFB56002, all necessary device configuration data to select the master devices (CIF cards) and the information about device specific data blocks which are used during runtime.

The program design allows an easy integration of the CIF-WinAC driver into an existing STEP7 project.

SFB65001 and SFB65002 are accessed via multi instances, so other program parts are also able to uses these SFBs.

All driver data blocks are configured in DB_CONFIG, so they can be renamed and only the information about the used data blocks in DB_CONFIG must be changed.

© Hilscher, 2009

7.1 Program Structure

Organization Block	Function	
OB100	Network1	Reset INIT_OK flag (located in DB_CONFIG) U "DB_CONFIG".INIT_OK R "DB_CONFIG".INIT_OK
	Network2	Start CIF driver and initialize all devices CALL "CIFACDriverInit" , "DB_CONFIG" ERROR:="CIFAC_Status_Word"
	Network3	Stop on initialization errors (CIFAC_Status_Word <> 0)
OB1	Network1	Read inputs CALL "CIFACReadInput" , "DB_CIFACReadInput" DB_CONFIG:="DB_CONFIG" DEV :=0 // process all devices ERROR :="CIFAC_Status_Word"
	Network2	Stop PLC on read input error (CIFAC_Status_Word <> 0)
	USER PROGRAMM	
	Network (n)	Write outputs to bus CALL "CIFACWriteOutput" , "DB_CIFACWriteOutput" DB_CONFIG:="DB_CONFIG" DEV :=0 // process all devices ERROR :="CIFAC_Status_Word"
	Network (n+1)	Stop PLC on write output error (CIFAC_Status_Word <> 0)

Table 14: Program Structure

© Hilscher, 2009

7.2 Data Organization

DB_CONFIG is the global configuration data block and the only one which has to be changed.

CIFAC_Status_Word is the global error value for all FBs.

Each device uses at least three data blocks
DB_I_MAP, **DB_Q_MAP**, **DB_SLAVE_STATE_MAP**.

Other data blocks like **DB_SLAVE_CFG**, **DB_DEVICE_INFO**, **DB_MSG_WRITE** and **DB_MSG_READ** are predefined for specific functions.

NOTE: Do not change the structure and length of any data block, except the **DB_I_MAP** and **DB_Q_MAP** blocks.

Global Data	**Definition**	**Description**
MW 60	CIFAC_Status_Word	Global state/error word for all FBs
Data Block	**Definition**	**Description**
DB2	DB_CONFIG	Global configuration data block
DB3,DB4,DB5,DB6	Instance DB	Instance DBs for function blocks FB3,FB4,FB5,FB6
DB1000	DB_SLAVE_CFG	Standard data block to read a slave configuration
DB1001	DB_DEVICE_INFO	Standard data block to a device information
DB1002	DB_MSG_WRITE	Data block to write a message to the CIF Device
DB1003	DB_MSG_READ	Data block to read a message from the CIF Device
Device 1		
DB1101	DB_IMAP_DEV_1	Process data image for INPUT data
DB1102	DB_QMAP_DEV_1	Process data image for OUTPUT data
DB1103	DB_STATEMAP_DEV_1	Device 1 slave state map
Device 2		
DB1201	DB_IMAP_DEV_2	Process data image for INPUT data
DB1202	DB_QMAP_DEV_2	Process data image for OUTPUT data
DB1203	DB_STATEMAP_DEV_2	Device 2 slave state map
Device 3		
DB1301	DB_IMAP_DEV_3	Process data image for INPUT data
DB1302	DB_QMAP_DEV_3	Process data image for OUTPUT data
DB1303	DB_STATEMAP_DEV_3>	Device 3 slave state map
Device 4		
DB1401	DB_IMAP_DEV_4	Process data image for INPUT data
DB1402	DB_QMAP_DEV_4	Process data image for OUTPUT data
DB1403	DB_STATEMAP_DEV_4	Device 4 slave state map

Table 15: Data Organization

© Hilscher, 2009

7.3 Function Blocks

Overview:

Function Block	Definition	Cmd	Description
FB2	CIFACDriverInit	1	-CIFACDriver initialization and initialization of the specified CIF devices - Start communication of the fieldbus system if the device successfully initialized
FB3	CIFACReadInput	3	- Read INPUT data from CIF devices - Transfer slave state information to the slave state map
FB4	CIFACWriteOutput	4	- Write OUTPUT data to CIF devices - Transfer slave state information to the slave state map
FB5	CIFACReadSlaveConfig	2	Read a slave configuration information into a given DB
FB6	CIFACReadDeviceInfo	12	Reads global information about a device into a given DB
FB7	CIFACWriteMsg	7	Write a messages to the CIF device
FB8	CIFACReadMsg	8	Read a messages from the CIF device

Table 16: Function Blocks

7.3.1 FB2: CIFACDriverInit

Description: Initialize all devices, configured in **DB_CONFIG**.

Direction	Type	Parameter	Description
out	WORD	ERROR	Error return

Table 17: FB2: CIFACDriverInit

7.3.2 FB3: CIFACReadInput

Description: Read the input process data from the CIF hardware and store the data into the input image map (**DB_I_MAP_DEVICEx**) of the device.

Direction	Type	Parameter	Description
in	BLOCK_DB	DB_CONFIG	Global configuration data block
in	INT	DEV	Device number 0 = process all devices 1..4 = process specific device
out	WORD	ERROR	Error return

Table 18: FB3: CIFACReadInput

© Hilscher, 2009

7.3.3 FB4: CIFACWriteOutput

Description: Write the output process data to the CIF hardware used the data from the output image map (**DB_Q_MAP_DEVICEx**) of the device.

Direction	Type	Parameter	Description
in	BLOCK_DB	DB_CONFIG	Global configuration data block
in	INT	DEV	Device number 0 = process all devices 1..4 = process specific device
out	WORD	ERROR	Error return

Table 19: FB4: CIFACWriteOutput

7.3.4 FB5: CIFACReadSlaveConfig

Description: Read the configuration data of a specific slave device. Use the data block **DB_SLAVE_CFG** to store the data.

Direction	Type	Parameter	Description
in	BLOCK_DB	DB_CONFIG	Global data block
in	INT	DEV	Device number 1..4
in	INT	SLAVE_ADDRESS	Slave address to read the configuration (0..127)
in	BLOCK_DB	SLAVE_CONFIG_DB	Data block for slave data. Pass the data block number of the predefined data block DB_SLAVE_CFG.
out	WORD	ERROR	Error return

Table 20: FB5: CIFACReadSlaveConfig

7.3.5 FB6: CIFACReadDeviceInfo

Description: Read various information from a specific CIF device. Use the data block **DB_DEVICE_INFO** to store the data.

Direction	Type	Parameter	Description
in	BLOCK_DB	DB_CONFIG	Global data block
in	INT	DEV	Device number 1..4
in	BLOCK_DB	DEVICE_INFO_DB	Data block for device information. Pass the data block number of the predefined data block DB_DEVICE_INFO.
out	WORD	ERROR	Error return

Table 21: FB6: CIFACReadDeviceInfo

© Hilscher, 2009

7.3.6 FB7: CIFACWriteMsg

Description: Writes a message to the CIF hardware (for asynchronous fieldbus protocol services). Use the data block **DB_MSG_WRITE** to store the data.

Direction	Type	Parameter	Description
in	BLOCK_DB	DB_CONFIG	Global data block
in	INT	DEV	Device number 1..4
in	BLOCK_DB	MSG_WRITE_DB	Data block for message writing. Pass the data block number of the predefined data block DB_MSG_WRITE.
out	WORD	ERROR	Error return

Table 22: FB7: CIFACWriteMsg

7.3.7 FB8: CIFACReadMsg

Description: Reads a message from the CIF hardware (for asynchronous fieldbus protocol services). Use the data block **DB_MSG_READ** to store the data.

Direction	Type	Parameter	Description
in	BLOCK_DB	DB_CONFIG	Global data block
in	INT	DEV	Device number 1..4
in	BLOCK_DB	MSG_READ_DB	Data block for message reading. Pass the data block number of the predefined data block DB_MSG_READ.
out	WORD	ERROR	Error return

Table 23: FB8: CIFACReadMsg

© Hilscher, 2009

7.4 Error Handling

The error handling is done by the global error word **CIFAC_Status_Word** (MW 60) and an error value (**DeviceError**) for each devices located in **DB_CONFIG**.

All FBs expecting an error word, where the global word **CIFAC_Status_Word** is used. During device initialization in FB2 the global error can hold all error values. Later during runtime, **CIFAC_Status_Word** only signals device specific errors with the value 1 and the error can be found in ***DB_CONFIG.DEVICE[x].DeviceError***.

Possible error values:

Name	Values
CIFAC_Status_Word	0 = No error 1 = Device specific error. Check the **DeviceError** entry of all devices >1 = Global error (see error table)

© Hilscher, 2009

8 Error Numbers

8.1 WinAC Basis Error List

WinAC Basis		
Value	**Name**	**Description**
SFB65001		
0	NO_ERRORS	Success
0x807F	ERROR_INTERNAL	An Internal error occurred
0x8001	E_EXCEPTION	An exception occurred
0x8102	E_CLSID_FAILED	The call CLSIDFromProgID failed
0x8103	E_COINITIALIZE_FAILED	The call to CoInitializeEx failed
WinAC Basis SFB65002		
0	NO_ERRORS	Success
0x807F	ERROR_INTERNAL	An Internal error occurred
0x8001	E_EXCEPTION	An exception occurred
0x8002	E_NO_VALID_INPUT	Input: the ANY pointer is invalid
0x8003	E_INPUT_RANGE_INVALID	Input: the ANY pointer range is invalid
0x8004	E_NO_VALID_OUTPUT	Output: the ANY pointer is invalid
0x8005	E_OUTPUT_RANGE_INVALID	Output: the ANY pointer range is invalid
0x8006	E_OUTPUT_OVERFLOW	More bytes are written into the buffer by the COM object than allocated
0x8007	E_NOT_INITILIZED	COM system has not been initialized: no previous call to SFB65001(“CREA_COM”)
0x8008	E_HANDLE_OUT_OF_RANGE	The supplied handle value does not correspond to a valid COM object

Table 24: Error Numbers – WinAC Basis Error List

© Hilscher, 2009

8.2 WinAC RTX Error List

WinAC RTX		
Value	**Name**	**Description**
SFB65001		
0	NO_ERRORS	Success
0x807F	ERROR_INTERNAL	An internal error occurred.
0x8001	E_EXCEPTION	An exception occurred.
0x8002	E_NO_VALID_INPUT	Input: the ANY pointer is invalid.
0x8003	E_INPUT_RANGE_INVALID	Input: the ANY pointer range is invalid.
0x8004	E_NO_VALID_OUTPUT	Output: the ANY pointer is invalid.
0x8005	E_OUTPUT_RANGE_INVALID	Output: the ANY pointer range is invalid.
0x8006	E_OUTPUT_OVERFLOW	More bytes were written into the output buffer by the extension object than were allocated.
0x8007	E_NOT_INITIALIZED	ODK system has not been initialized: no previous call to SFB65001 (CREA_COM).
0x8008	E_HANDLE_OUT_OF_RANGE	The supplied handle value does not correspond to a valid extension object.
0x8009	E_INPUT_OVERFLOW	More bytes were written into the input buffer by the extension object than were allocated.
SFB65002		
0	NO_ERRORS	Success
0x807F	ERROR_INTERNAL	An internal error occurred.
0x8001	E_EXCEPTION	An exception occurred.
0x8002	E_NO_VALID_INPUT	Input: the ANY pointer is invalid.
0x8003	E_INPUT_RANGE_INVALID	Input: the ANY pointer range is invalid.
0x8004	E_NO_VALID_OUTPUT	Output: the ANY pointer is invalid.
0x8005	E_OUTPUT_RANGE_INVALID	Output: the ANY pointer range is invalid.
0x8006	E_OUTPUT_OVERFLOW	More bytes were written into the output buffer by the extension object than were allocated.
0x8007	E_NOT_INITIALIZED	ODK system has not been initialized: no previous call to SFB65001 (CREA_COM).
0x8008	E_HANDLE_OUT_OF_RANGE	The supplied handle value does not correspond to a valid extension object.
0x8009	E_INPUT_OVERFLOW	More bytes were written into the input buffer by the extension object than were allocated.

Table 25: Error Numbers – WinAC RTX Error List

© Hilscher, 2009

8.3 CIF-WinAC Driver Error List

Value	Name	Description
0	NO_ERROR	Success
1	GLOBAL_ERROR	Check device error values
Errors from underlying hardware driver (CIF Device Driver / CIF RTX driver)		
1000	DRIVER_ALREADY_OPENED	
1001	DRIVER_DLL_NAME_INVALID	
1002	DRIVER_DLL_NOT_FOUND	
1003	DRIVER_FUNCTION_MISSING	
1004	DRIVER_FUNCTION_FAILED	
1005	DRIVER_DEVICE_NAME_INVALID	
1006	DRIVER_DEVICE_TYPE_UNKNOWN	
1007	DRIVER_FW_FILENAME_NOT_FOUND	
1008	DRIVER_DEVICE_NUMBER_INVALID	
1009	DRIVER_SERIAL_NUMBER_INVALID	
Errors from the CIFWinAC driver		
0x8510	E_COMMAND_NOT_IMPLEMENTED	Invalid subcommand number
0x8520	E_COMMAND_UNKNOWN_DRIVER	Driver unknown
0x8521	E_COMMAND_UNKNOWN_DEVICE	Device unknown
0x8522	E_COMMAND_MEMORY_NOT_AVAILABLE	Internal memory not available
0x8523	E_COMMAND_UNKNOWN_MODE	Mode parameter unknown
0x9030	E_COMMAND_DEVICE_ERROR	Global device error
0x9031	E_COMMAND_INVALID_INPUT_SIZE	Input size of defined DB invalid
0x9032	E_COMMAND_INVALID_OUTPUT_SIZE	Output size of defined DB invalid
0x9033	E_COMMAND_INVALID_DEVICE_NUMBER	Invalid device number given
0x9034	E_COMMAND_INVALID_DEVICE_HANDLE	Invalid device handle found
0x9035	E_COMMAND_DEVICE_ALREADY_ACTIVE	Device already active
0x9040	E_COMMAND_CFG_INP_DIRECTION_ERROR	Device configuration input direction error
0x9041	E_COMMAND_CFG_INP_DATATYPE_ERROR	Device configuration input data type error
0x9042	E_COMMAND_CFG_OUT_DIRECTION_ERROR	Device configuration output direction error
0x9043	E_COMMAND_CFG_OUT_DATATYPE_ERROR	Device configuration output data type error

Table 26: Error Numbers – WinAC Driver Error List

© Hilscher, 2009

8.4 CIF Device Driver Error List

The column *Hint* shows if there is additional error information available. If 'Yes' then see section *Additional Error Information*, which is the next section.

Value	Name	Description	Hint
0	DRV_NO_ERROR	no error	
-1	DRV_BOARD_NOT_INITIALIZED	DRIVER Board not initialized	yes
-2	DRV_INIT_STATE_ERROR	DRIVER Error in internal init state	
-3	DRV_READ_STATE_ERROR	DRIVER Error in internal read state	
-4	DRV_CMD_ACTIVE	DRIVER Command on this channel is active	
-5	DRV_PARAMETER_UNKNOWN	DRIVER Unknown parameter in function occurred	
-6	DRV_WRONG_DRIVER_VERSION	DRIVER Version is incompatible with DLL	
-7	DRV_PCI_SET_CONFIG_MODE	DRIVER Error during PCI set run mode	
-8	DRV_PCI_READ_DPM_LENGTH	DRIVER Could not read PCI dual port memory length	
-9	DRV_PCI_SET_RUN_MODE	DRIVER Error during PCI set run mode	
-10	DRV_DEV_DPM_ACCESS_ERROR	DEVICE Dual port ram not accessible (board not found)	yes
-11	DRV_DEV_NOT_READY	DEVICE Not ready (ready flag failed)	yes
-12	DRV_DEV_NOT_RUNNING	DEVICE Not running (running flag failed)	yes
-13	DRV_DEV_WATCHDOG_FAILED	DEVICE Watchdog test failed	yes
-14	DRV_DEV_OS_VERSION_ERROR	DEVICE Signals wrong OS version	yes
-15	DRV_DEV_SYSERR	DEVICE Error in dual port flags	
-16	DRV_DEV_MAILBOX_FULL	DEVICE Send mailbox is full	
-17	DRV_DEV_PUT_TIMEOUT	DEVICE PutMessage timeout	yes
-18	DRV_DEV_GET_TIMEOUT	DEVICE GetMessage timeout	yes
-19	DRV_DEV_GET_NO_MESSAGE	DEVICE No message available	
-20	DRV_DEV_RESET_TIMEOUT	DEVICE RESET command timeout	yes
-21	DRV_DEV_NO_COM_FLAG	DEVICE COM-flag not set	yes
-22	DRV_DEV_EXCHANGE_FAILED	DEVICE IO data exchange failed	
-23	DRV_DEV_EXCHANGE_TIMEOUT	DEVICE IO data exchange timeout	yes
-24	DRV_DEV_COM_MODE_UNKNOWN	DEVICE IO data mode unknown	
-25	DRV_DEV_FUNCTION_FAILED	DEVICE Function call failed	
-26	DRV_DEV_DPMSIZE_MISMATCH	DEVICE DPM size differs from configuration	
-27	DRV_DEV_STATE_MODE_UNKNOWN	DEVICE State mode unknown	
-30	DRV_USR_OPEN_ERROR	USER Driver not opened (device driver not loaded)	yes
-31	DRV_USR_INIT_DRV_ERROR	USER Can't connect with device	
-32	DRV_USR_NOT_INITIALIZED	USER Board not initialized (DevInitBoard not called)	
-33	DRV_USR_COMM_ERR	USER IOCTRL function failed	
-34	DRV_USR_DEV_NUMBER_INVALID	USER Parameter DeviceNumber invalid	
-35	DRV_USR_INFO_AREA_INVALID	USER Parameter InfoArea unknown	
-36	DRV_USR_NUMBER_INVALID	USER Parameter Number invalid	
-37	DRV_USR_MODE_INVALID	USER Parameter Mode invalid	
-38	DRV_USR_MSG_BUF_NULL_PTR	USER NULL pointer assignment	
-39	DRV_USR_MSG_BUF_TOO_SHORT	USER Message buffer too short	
-40	DRV_USR_SIZE_INVALID	USER Parameter Size invalid	
-42	DRV_USR_SIZE_ZERO	USER Parameter Size with zero length	
-43	DRV_USR_SIZE_TOO_LONG	USER Parameter Size too long	

© Hilscher, 2009

Value	Name	Description	Hint
-44	DRV_USR_DEV_PTR_NULL	USER Device address null pointer	
-45	DRV_USR_BUF_PTR_NULL	USER Pointer to buffer is a null pointer	
-46	DRV_USR_SENDSIZE_TOO_LONG	USER Parameter SendSize too long	
-47	DRV_USR_RECVSIZE_TOO_LONG	USER Parameter ReceiveSize too long	
-48	DRV_USR_SENDBUF_PTR_NULL	USER Pointer to send buffer is a null pointer	
-49	DRV_USR_RECVBUF_PTR_NULL	USER Pointer to receive buffer is a null pointer	
-50	DRV_DMA_TIMEOUT_CH4	DMA read IO timeout	
-51	DRV_DMA_TIMEOUT_CH5	DMA write IO timeout	
-52	DRV_DMA_TIMEOUT_CH6	DMA PCI transfer timeout	
-53	DRV_DMA_TIMEOUT_CH7	DMA download timeout	
-54	DRV_DMA_INSUFF_RES_MEM	DMA Memory allocation error	
-70	DRV_ERR_ERROR	DRIVER General error	
-71	DRV_DMA_ERROR	DRIVER General DMA error	
-72	DRV_BATT_ERROR	DRIVER Battery error	
-73	DRV_PWF_ERROR	DRIVER Power failed error	
-80	DRV_USR_DRIVER_UNKNOWN	USER driver unknown	
-81	DRV_USR_DEVICE_NAME_INVALID	USER device name invalid	
-82	DRV_USR_DEVICE_NAME_UKNOWN	USER device name unknown	
-83	DRV_USR_DEVICE_FUNC_NOTIMPL	USER device function not implemented	
-100	DRV_USR_FILE_OPEN_FAILED	USER file not opened	
-101	DRV_USR_FILE_SIZE_ZERO	USER file size zero	
-102	DRV_USR_FILE_NO_MEMORY	USER not enough memory to load file	
-103	DRV_USR_FILE_READ_FAILED	USER file read failed	
-104	DRV_USR_INVALID_FILETYPE	USER file type invalid	
-105	DRV_USR_FILENAME_INVALID	USER file name not valid	
-110	DRV_FW_FILE_OPEN_FAILED	USER firmware file not opened	
-111	DRV_FW_FILE_SIZE_ZERO	USER firmware file size zero	
-112	DRV_FW_FILE_NO_MEMORY	USER not enough memory to load firmware file	
-113	DRV_FW_FILE_READ_FAILED	USER firmware file read failed	
-114	DRV_FW_INVALID_FILETYPE	USER firmware file type invalid	
-115	DRV_FW_FILENAME_INVALID	USER firmware file name not valid	
-116	DRV_FW_DOWNLOAD_ERROR	USER firmware file download error	
-117	DRV_FW_FILENAME_NOT_FOUND	USER firmware file not found in the internal table	
-118	DRV_FW_BOOTLOADER_ACTIVE	USER firmware file BOOTLOADER active	
-119	DRV_FW_NO_FILE_PATH	USER firmware file not file path	
-120	DRV_CF_FILE_OPEN_FAILED	USER configuration file not opened	
-121	DRV_CF_FILE_SIZE_ZERO	USER configuration file size zero	
-122	DRV_CF_FILE_NO_MEMORY	USER not enough memory to load configuration file	
-123	DRV_CF_FILE_READ_FAILED	USER configuration file read failed	
-124	DRV_CF_INVALID_FILETYPE	USER configuration file type invalid	
-125	DRV_CF_FILENAME_INVALID	USER configuration file name not valid	
-126	DRV_CF_DOWNLOAD_ERROR	USER configuration file download error	
-127	DRV_CF_FILE_NO_SEGMENT	USER no flash segment in the configuration file	
-128	DRV_CF_DIFFERS_FROM_DBM	USER configuration file differs from database	
-131	DRV_DBM_SIZE_ZERO	USER database size zero	
-132	DRV_DBM_NO_MEMORY	USER not enough memory to upload database	
-133	DRV_DBM_READ_FAILED	USER database read failed	

© Hilscher, 2009

Value	Name	Description	Hint
-136	DRV_DBM_NO_FLASH_SEGMENT	USER database segment unknown	
-150	DEV_CF_INVALID_DESCRIPT_VERSION	CONFIG version of the Descript table invalid	
-151	DEV_CF_INVALID_INPUT_OFFSET	CONFIG input offset is invalid	
-152	DEV_CF_NO_INPUT_SIZE	CONFIG input size is 0	
-153	DEV_CF_MISMATCH_INPUT_SIZE	CONFIG input size does not match configuration	
-154	DEV_CF_INVALID_OUTPUT_OFFSET	CONFIG invalid output offset	
-155	DEV_CF_NO_OUTPUT_SIZE	CONFIG output size is 0	
-156	DEV_CF_MISMATCH_OUTPUT_SIZE	CONFIG output size does not match configuration	
-157	DEV_CF_STN_NOT_CONFIGURED	CONFIG Station not configured	
-158	DEV_CF_CANNOT_GET_STN_CONFIG	CONFIG cannot get the Station configuration	
-159	DEV_CF_MODULE_DEF_MISSING	CONFIG Module definition is missing	
-160	DEV_CF_MISMATCH_EMPTY_SLOT	CONFIG empty slot mismatch	
-161	DEV_CF_MISMATCH_INPUT_OFFSET	CONFIG input offset mismatch	
-162	DEV_CF_MISMATCH_OUTPUT_OFFSET	CONFIG output offset mismatch	
-163	DEV_CF_MISMATCH_DATA_TYPE	CONFIG data type mismatch	
-164	DEV_CF_MODULE_DEF_MISSING_NO_SI	CONFIG Module definition is missing,(no Slot/Idx)	
>=1000	RCS_ERROR	Board operation system errors will be passed with this offset (e.g. error 1234 means RCS error 234). Only if a ready fault occurred during board initialization.	

Table 27: Error Numbers – CIF Device Driver Error List

© Hilscher, 2009

8.4.1 Additional Error Information

This section contains more information about possible reasons to certain error numbers.

Error: -1
The communication board is not initialized by the driver.
No or wrong configuration found for the given board.
- Check the driver configuration
- Driver function used without calling DevOpenDriver() first

Error: -6
The device driver version does not corresponds to the driver DLL version. From version V1.200 the internal command structure between DLL and driver has changed.
- Make sure to use the same version of the device driver and the driver DLL

Error: -10
Dual ported RAM (DPM) not accessible / no hardware found.
This error occurs, when the driver is not able to read or write to the DPM
- Check the BIOS setting of the PC
- Memory address conflict with other PC components, try another memory address
- Check the driver configuration for this board
- Check the jumper setting of the board

Error: -11
Board is not ready.
This is a general error, the board has a hardware malfunction.

Error: -12
At least one task is not initialized. The board is ready but not all tasks are running.
- No data base is loaded into the device
- Wrong parameter that causes that a task can't initialize. Use ComPro menu *Online-task-version*.

Error: -14
No license code found on the communication board.
- Device has no license for the used operating system or customer software.
- No firmware or no data base on the device loaded.

Error: -17
No message could be send during the timeout period given in the `DevPutMessage()` function.
- Using device interrupts
Wrong or no interrupt selected. Check interrupt on the device and in driver registration. They have to be the same!. Interrupt already used by an other PC component.
- Device internal segment buffer full
`PutMessage()` function not possible, because all segments on the device are in use. This error occurs, when only `PutMessage()` is used but not `GetMessage()`.
- HOST flag not set for the device
No messages are taken by the device. Use DevSetHostState() to signal a board an application is available.

Error: -18
No message received during the timeout period given in the `DevGetMessage()` function.
- Using device interrupts

© Hilscher, 2009

Wrong or no interrupt selected. Check interrupt on the device and in driver registration. They have to be the same!. Interrupt already used by an other PC component.
- The used protocol on the device needs longer than the timeout period given in the `DevGetMessage()` function

Error: -20
The device needs longer than the timeout period given in the `DevReset()` function
- Using device interrupts
This error occurs when for example interrupt 9 is set in the driver registration but no or a wrong interrupt is jumpered on the device (=device in pollmode).
Interrupt already used by an other PC component.
- The timeout period can differ between fieldbus protocols

Error: -21
The device can not reach communication state.
- Device not connected to the fieldbus
- No station found on the fieldbus
- Wrong configuration on the device

Error: -23
The device needs longer than the timeout period given in the `DevExchangeIO()` function.
- Using device interrupts
Wrong or no interrupt selected. Check interrupt on the device and in driver registration. They have to be the same!. Interrupt already used by an other PC component.

Error: -30
The device driver could not be opened.
- Device driver not installed
- Wrong parameters in the driver configuration
If the driver finds invalid parameters for a communication board and no other boards with valid parameters are available, the driver will not be loaded.

Error: -33
A driver function could not be called. This is an internal error between the device driver and the DLL.
- Make sure to use a device driver and a DLL with the same version.
- An incompatible old driver DLL is used.

© Hilscher, 2009

9 Appendix

9.1 List of Tables

Table 1: List of Revisions 5
Table 2: References to Additional Information 5
Table 3: CD Contents 8
Table 4: Requirements - WinAC Basis V3.x +SP1 / 4.1 10
Table 5: Requirements - WinAC RTX V3.x / 4.0 / V4.1 10
Table 6: Requirements - WinAC RTX 2005 10
Table 7: Requirements - WinAC RTX 2008 11
Table 8: Requirements - WinAC RTX 2009 11
Table 9: Supported CIF Cards and Fieldbus Systems (Part 1) 13
Table 10: Supported CIF Cards and Fieldbus Systems (Part 2) 14
Table 11: Driver Loading, Initialization and Removing – WinAC Basis 29
Table 12: Driver Loading, Initialization and Removing – WinAC RTX 29
Table 13: Driver Functions 30
Table 14: Program Structure 33
Table 15: Data Organization 34
Table 16: Function Blocks 35
Table 17: FB2: CIFACDriverInit 35
Table 18: FB3: CIFACReadInput 35
Table 19: FB4: CIFACWriteOutput 36
Table 20: FB5: CIFACReadSlaveConfig 36
Table 21: FB6: CIFACReadDeviceInfo 36
Table 22: FB7: CIFACWriteMsg 37
Table 23: FB8: CIFACReadMsg 37
Table 24: Error Numbers – WinAC Basis Error List 39
Table 25: Error Numbers – WinAC RTX Error List 40
Table 26: Error Numbers – WinAC Driver Error List 41
Table 27: Error Numbers – CIF Device Driver Error List 44

9.2 List of Figures

Figure 1: STEP7 Hardware Configuration 17
Figure 2: STEP7 Project View 18
Figure 3: STEP7 Declaration View of DB_CONFIG 19
Figure 4: STEP7 Structure of a RCS Message 22

© Hilscher, 2009

9.3 Contacts

Headquarters

Germany
Hilscher Gesellschaft für
Systemautomation mbH
Rheinstrasse 15
65795 Hattersheim
Phone: +49 (0) 6190 9907-0
Fax: +49 (0) 6190 9907-50
E-Mail: info@hilscher.com

Support
Phone: +49 (0) 6190 9907-99
E-Mail: de.support@hilscher.com

Subsidiaries

China
Hilscher Ges.f.Systemaut. mbH
Shanghai Representative Office
200010 Shanghai
Phone: +86 (0) 21-6355-5161
E-Mail: info@hilscher.cn

Support
Phone: +86 (0) 21-6355-5161
E-Mail: cn.support@hilscher.com

France
Hilscher France S.a.r.l.
69500 Bron
Phone: +33 (0) 4 72 37 98 40
E-Mail: info@hilscher.fr

Support
Phone: +33 (0) 4 72 37 98 40
E-Mail: fr.support@hilscher.com

India
Hilscher India Pvt. Ltd.
New Delhi - 110 025
Phone: +91 9810269248
E-Mail: info@hilscher.in

Italy
Hilscher Italia srl
20090 Vimodrone (MI)
Phone: +39 02 25007068
E-Mail: info@hilscher.it

Support
Phone: +39/02 25007068
E-Mail: it.support@hilscher.com

Japan
Hilscher Japan KK
Tokyo, 160-0022
Phone: +81 (0) 3-5362-0521
E-Mail: info@hilscher.jp

Support
Phone: +81 (0) 3-5362-0521
E-Mail: jp.support@hilscher.com

Switzerland
Hilscher Swiss GmbH
4500 Solothurn
Phone: +41 (0) 32 623 6633
E-Mail: info@hilscher.ch

Support
Phone: +49 (0) 6190 9907-99
E-Mail: ch.support@hilscher.com

USA
Hilscher North America, Inc.
Lisle, IL 60532
Phone: +1 630-505-5301
E-Mail: info@hilscher.us

Support
Phone: +1 630-505-5301
E-Mail: us.support@hilscher.com

© Hilscher, 2009